Programmeerimise algkursus

I200

Loengu sisu

- Meeldetuletus: kompileerimine, käivitamine, programmi struktuur
- Muutujad
- Andmetüübid
- Lihtsad näited: rehkendamine, trükkimine
- Veidi keerukamad näited: tsüklid

Programmi kirjutamise etapid:

- **getting the program text into the computer,**
- **compiling the program, and**
- **running the compiled program.**

Final step - running the program - either as

- **Application** - program running without a www browser

- Applet- program running in a www browser
- Servlet- program running in a (web) server

Java: APPLICATION

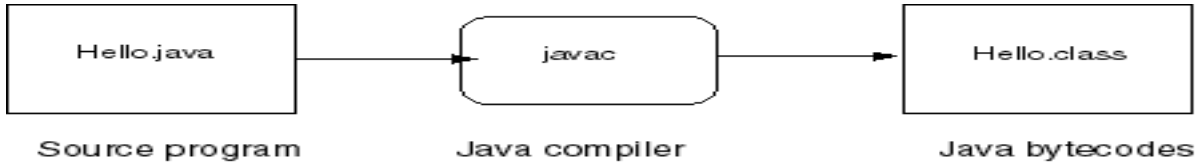
```
public class HelloWorld {

   // A program to display the message
   // "Hello World!" on standard output

   public static void main(String[] args) {
        System.out.println("Hello World!");
   }

}    // end of class HelloWorld
```

- kompileerime: **javac HelloWorld.java =>** HelloWorld.class
- paneme käsurealt käima: **java HelloWorld**

function called **main**, with a definition that takes the form:

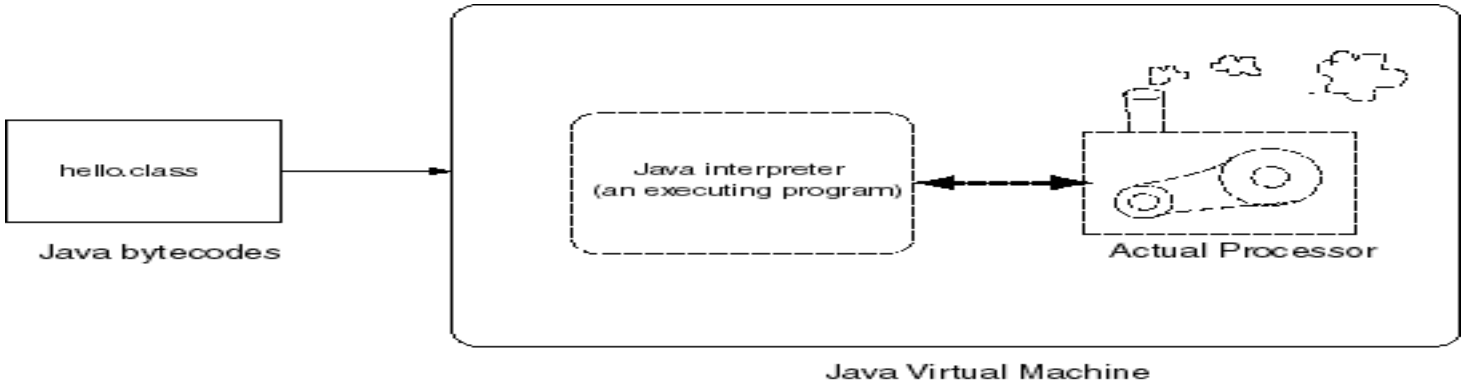
```
   public static void main(String[] args) {
      statements
   }
```

Compiler, bytecodes



Java Program Translation

Executing bytecodes



Java Bytecode Interpretation on a Virtual Machine

Program class structure

A program is defined by a public class that takes the form:

```
public class program-name {

  optional-variable-declarations-and-subroutines

  public static void main(String[] args) {
    statements
  }

  optional-variable-declarations-and-subroutines

}
```

Variables & primitive names

- `N n rate x15 a_long_name time_is_$ HelloWorld`

- `HelloWorld, helloworld, HELLOWORLD, and hElloWorLD are all distinct names`

- reserved words include: `class, public, static, if, else, while, and several dozen other words.`

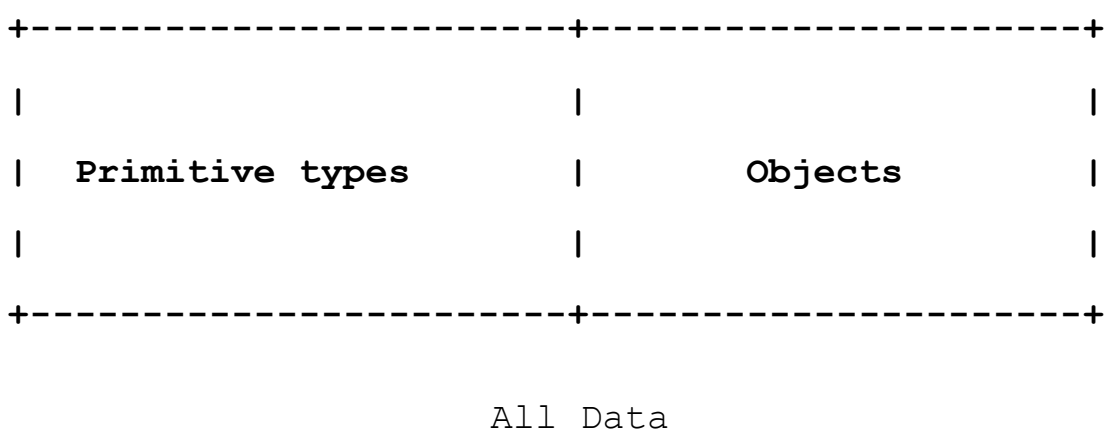
- An assignment statement takes the form:

  variable = expression;

```
rate = 0.07;
interest = rate * principal;
```

Data

- Java has *very many* data types built into it, and you (as a programmer) can create as many more as you want.
- However, other than the primitive data types, *all the other data in a Java program will be represented as an object*. So there is a fundamental split in the data a Java program deals with:



Primitive TYPES

- The primitive types are named:

```
byte, short, int, long,
float, double,
char,
boolean
```

- **short** corresponds to two bytes (16 bits). Variables of type short have values in the range -32768 to 32767.
- **int** corresponds to four bytes (32 bits). Variables of type int have values in the range -2147483648 to 2147483647.
- **long** corresponds to eight bytes (64 bits). Variables of type long have values in the range -92233720368547758O8

```
boolean result: rate > 0.05
```

String: "I said, \"Are you listening!\"\n"

Variable assignment

type-name variable-name;

or

type-name variable-name = expression;

```
int N;
double x;
double rate = 0.07;
char space = ' ';
```

You can create several variables in the same declaration, if you separate them by commas. For example:

```
double x,y;
char first = 'D',
     middle = 'J',
     last = 'E';
int i, j = 17;
```

Interest calculation

```
public class Interest {
 public static void main(String[] args) {
     // the value of the investment
     double principal = 17000;
     // the annual interest rate
     double rate = 0.07;
     // interest earned in one year
     double interest;

    interest = principal * rate;
    principal = principal + interest;

    System.out.print
     ("The interest earned is $");
    System.out.println(interest);
    System.out.print
      ("The value of the investment after one year is $");
    System.out.println(principal);

  } // end of main()
} // end of class Interest
```

Block

- The block is the simplest type of statement. Its purpose is simply to group a sequence of statements into a single statement. The format of a block is:

```
{
    statements
}
```

- Here are two examples of blocks:

```
{
  System.out.print("The answer is ");
  System.out.println(ans);
}
```

```
{  // This block exchanges the values of x and y
   int temp = x;  // declare temp and store x in it
   x = y;         // copy value of y into x
   y = temp;      // copy value of temp into y
}
```

While LOOP

- A while loop has the form:

```
while (boolean-expression)
    statement
```

- Since the statement can be, and usually is, a block, many while loops have the form:

```
while (boolean-expression) {
    statements
}
```

While LOOP

- Here is an example of a while loop that simply prints out the numbers 1, 2, 3, 4, 5:

```
int number = 1;
while ( number < 6 ) {
      System.out.println(number);
      number = number + 1;
}
System.out.println("Done!");
```

While LOOP

```
public class Interest2 {

  public static void main(String[] args) {

    double principal = 17000;
    double rate = 0.07;
     int years = 0;  // counts years

    while (years < 5) {
        double interest = principal * rate;
        principal = principal + interest;
        years = years + 1;
        System.out.print
       ("The value of the investment after ");
        System.out.print(years);
        System.out.print(" years is $");
        System.out.println(principal);
    } // end of while loop
  } // end of main()
} // end of class Interest2
```

IF statement

- An if statement has one of the forms:

```
if ( boolean-expression )
    statement
else
    statement
```

```
if ( boolean-expression )
    statement
```

- As usual, each of the **statement**'s in an if statement can be a block, so that an if statement often looks like:

```
if ( boolean-expression ) {
    statements
}
else {
    statements
}
if ( boolean-expression ) {
    statements

}
```

IF statement

```
if ( x > y ) {
  int temp = x;   // declare temp and store x
  x = y;          // copy value of y into x
  y = temp;       // copy value of temp into y
}
```

Finally, here is an example of an if statement that includes an `else part`.

```
if ( years > 1 ) {

   System.out.print

   ("The value of the investment after ");

   System.out.print(years);
   System.out.print(" years is $");
}
else {  // handle case for 1 year
   System.out.print
    ("The value of the investment after 1 year is $");
}  // end of if statement
System.out.println(principal);
```